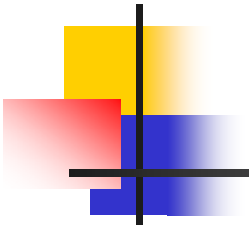


Finestre grafiche

MATLAB ha anche la possibilità di lavorare con delle finestre grafiche sulle quali si possono fare disegni bidimensionali o tridimensionali. Una finestra grafica viene aperta con il comando **figure** (in ambiente Windows anche con File-New-Figure). Si esegua:

```
>> figure
>> figure(n)
```

Il comando `plot`

L' istruzione per fare disegni bidimensionali è

`plot(x,y)`

dove x = (x1, ..., xn), y = (y1, ..., yn) sono vettori della stessa dimensione n

L'effetto è che sulla finestra grafica corrente (se non ci sono finestre grafiche ne viene aperta una) viene introdotto un sistema di coordinate cartesiane bidimensionale e si congiunge (x1, y1) con (x2, y2), (x2, y2) con (x3, y3), ..., (xn-1, yn-1) con (xn, yn) con una linea continua di colore blu.

Si provi

```
plot([0 2 3 4 6],[0 1 2 4 3])
```

Modalità di rappresentazione

Le modalità di rappresentazione dei punti possono essere cambiate dando un terzo argomento *stringa s* a plot che individua la modalità di rappresentazione.
L'istruzione plot nella sua forma completa è

plot(x,y,s)

Per vedere tutte le possibili modalità di rappresentazione si veda l'help in linea su plot.
Esempio:

```
>>plot([0 2 3 4 6],[0 1 2 4 3],'--')
>>plot([0 2 3 4 6],[0 1 2 4 3],'r+')
>>plot([0 2 3 4 6],[0 1 2 4 3],'ko')
```

Comandi utili

– grid: sovrappone al grafico un grigliato
– title: aggiunge un titolo del disegno
– xlabel: aggiunge una legenda per l'asse x
– ylabel: aggiunge una legenda per l'asse y
– axis: riscala gli assi del grafico
– clf : cancella il grafico corrente

```
>> xlabel('string')
>> title('string')
>> axis([xmin xmax ymin ymax]), axis square
```

Plot di funzioni reali

L'istruzione plot può fare più disegni sulla finestra grafica. Basta dare più coppie (x, y) (o terne(x, y, s)) come argomento a plot.

Ad esempio se si vuole tracciare in [0, 2] il grafico del seno e del coseno si possono usare le istruzioni

```
>>h=0.01;
>>x=0:h:2*pi;
>>y=sin(x);
>>z=cos(x);
>>plot(x,y,x,z)
```

Si noti come, automaticamente, MATLAB disegni i due grafici con colori diversi.

Gestione della finestra grafica

L'istruzione plot, prima di disegnare sulla finestra grafica corrente, cancella ogni disegno preesistente. Per evitare questo si usa il comando **hold on**. Dopo la sua esecuzione, sulla finestra grafica corrente viene mantenuto ogni disegno preesistente.

Volendo tracciare in [0, 2] il grafico del seno e del coseno si possono usare allora anche le istruzioni:

```
>>h=0.01;
>>x=0:h:2*pi;
>>y=sin(x);
>>plot(x,y);
>>hold on
>>z=cos(x);
>>plot(x,z,'r')
```

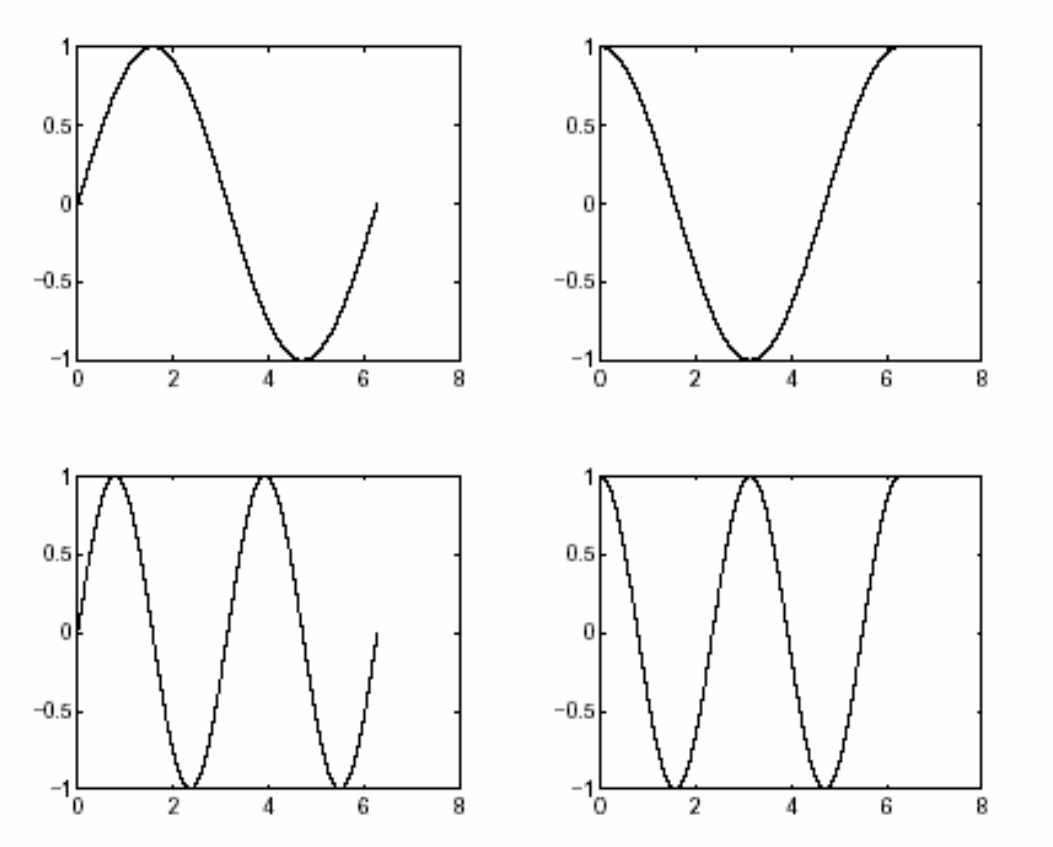
Per tornare alla situazione in cui vengono cancellati i disegni preesistenti si usa hold off.

Gestione della finestra grafica

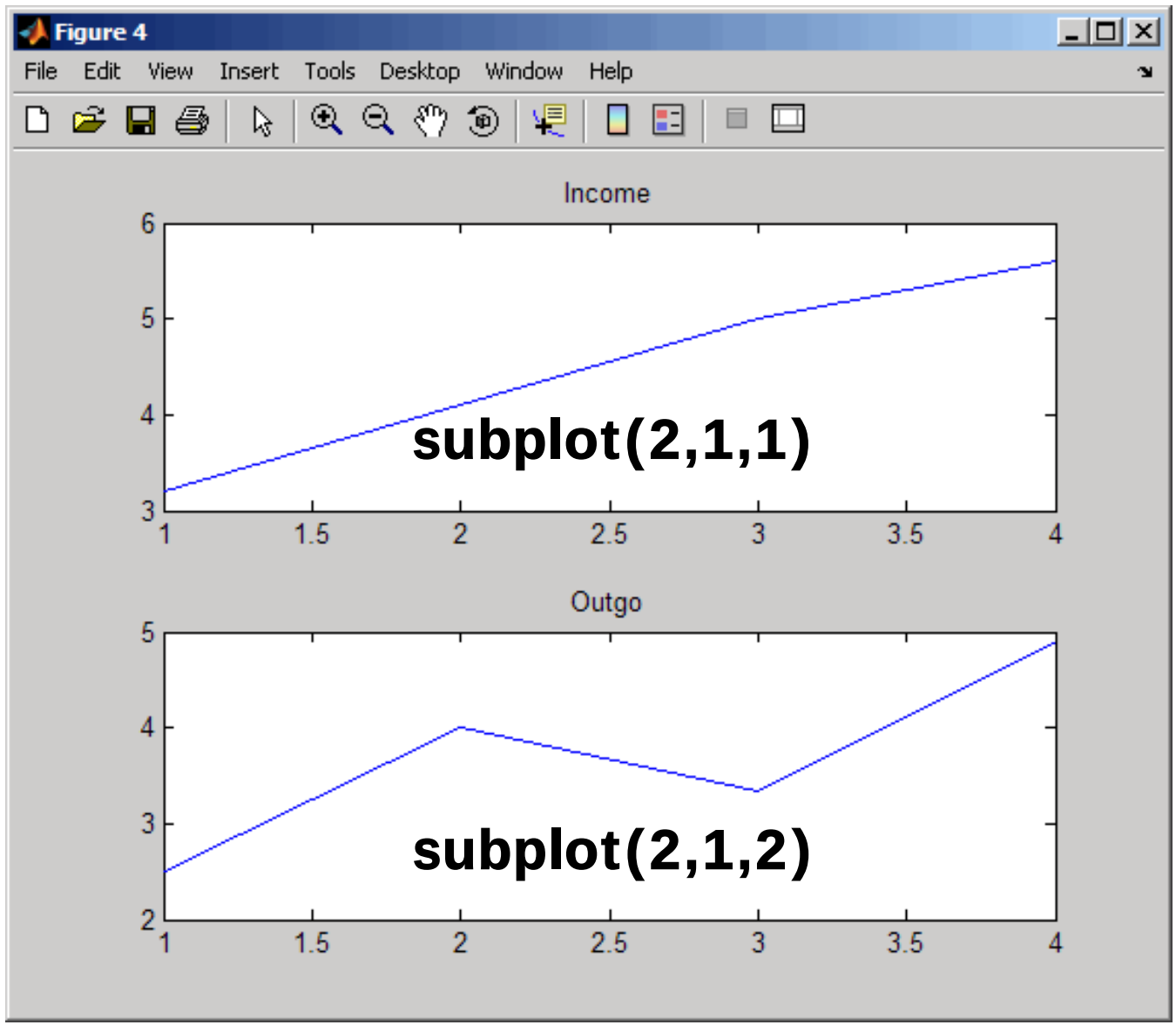
E’ possibile spezzare la finestra grafica corrente in una matrice [mxn] di sottofinestre grafiche. Il comando subplot(m,n,k) divide la finestra corrente (ne crea una se non esistono finestre grafiche) in una matrice [mxn] e fa diventare la k-esima, contata seguendo le righe, la finestra corrente.

```
>>x=0:h:2*pi;
>>subplot(2,2,1)
>>plot(x,sin(x))
>>subplot(2,2,2)
>>plot(x,cos(x))
>>subplot(2,2,3)
>>plot(x,sin(2*x))
>>subplot(2,2,4)
>>plot(x,cos(2*x))
```

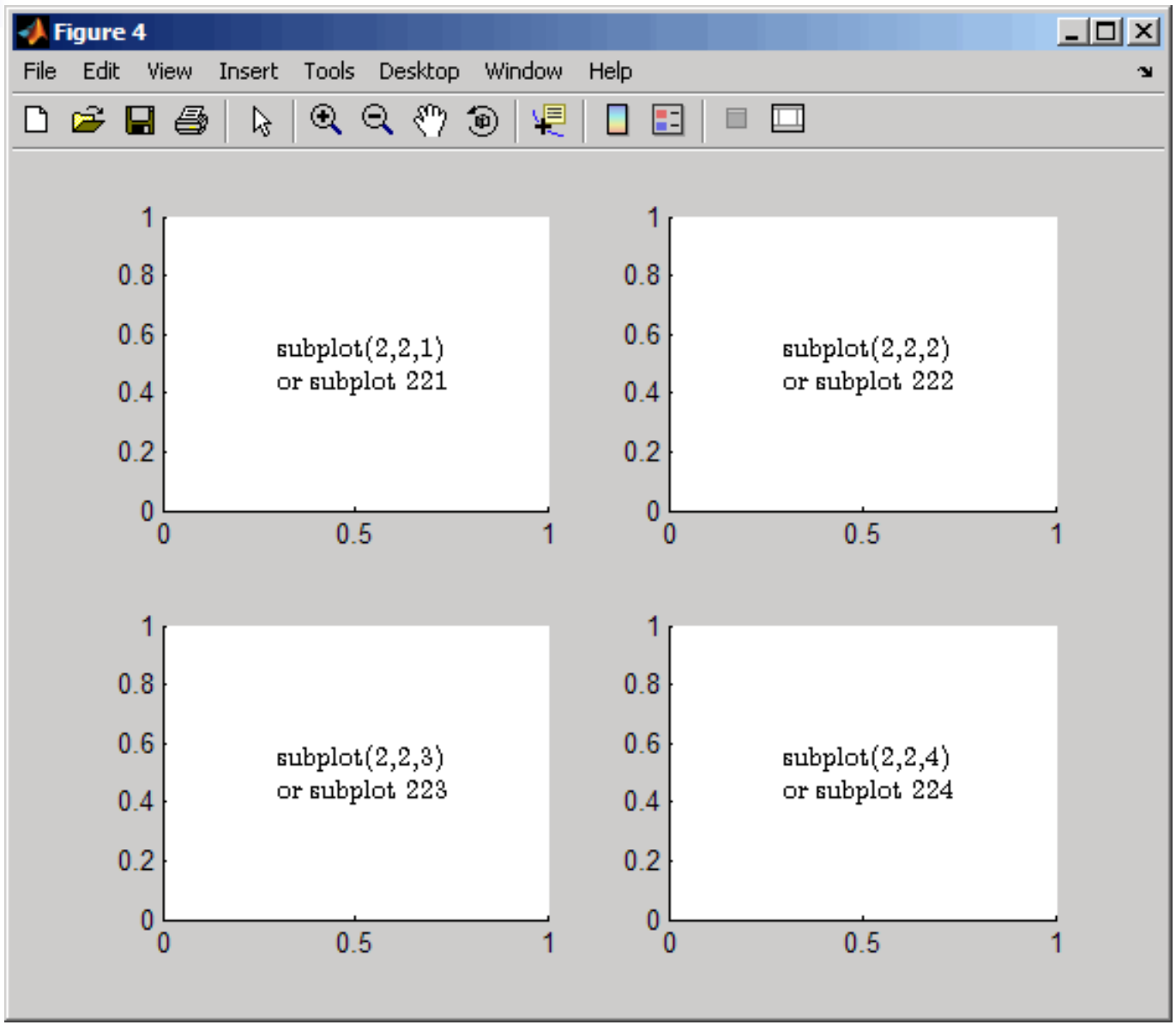
Gestione della finestra grafica



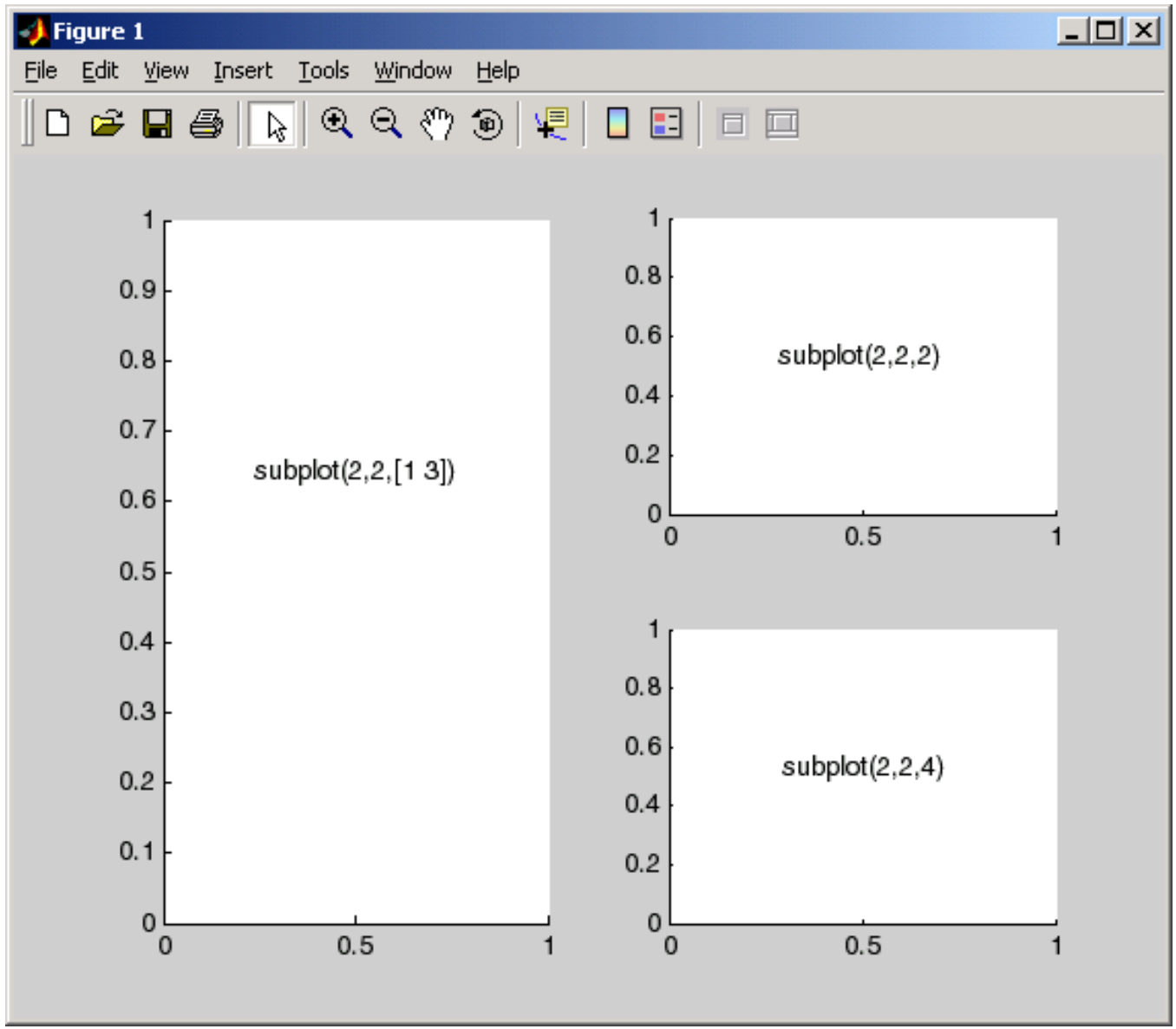
Gestione della finestra grafica



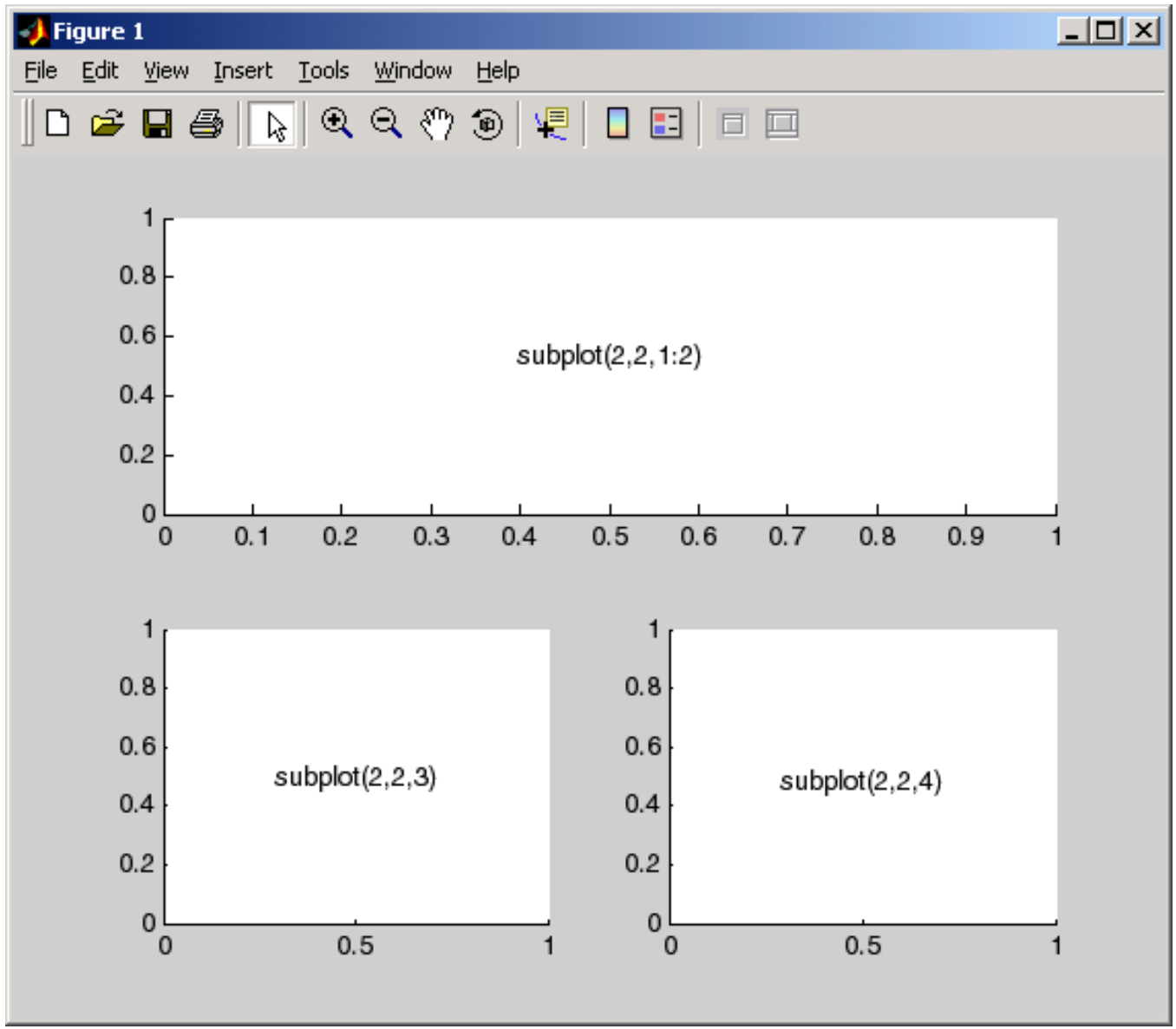
Gestione della finestra grafica



Gestione della finestra grafica



Gestione della finestra grafica



La programmazione

In Matlab si possono realizzare degli M-file, ovvero file di testo contenenti sequenze di comando e strutture di controllo che vengono interpretate dal Matlab.

I file, prodotti mediante un editor di testo devono essere salvati in un file con estensione .m, in una directory contenuta nel path.

Macro

Gli M-file di tipo macro operano sulle variabili contenute in memoria, e non esistono variabili locali. Contengono una serie di comandi che vengono automaticamente eseguiti quando si esegue la macro.
Per eseguire un M-file basta digitarne il nome (senza l'estensione) dalla riga di comando.

Per creare una macro:

1. aprire un file nuovo (emacs oppure File → New)
2. scrivere il codice della macro
3. salvare il file **`filename.m`**
4. eseguire da linea di comando >> filename + **`INVIO`**

Micro esempio di macro...

```
% ESEMPIO DI MACRO: calcola la matrice trasposta di
% una matrice A e ne visualizza l'output a schermo
(usare anche il comando disp).


Atrasp = A';

disp('la trasposta della matrice è')

Atrasp
```

Function

Matlab permette di definire funzioni utente.
Le funzioni vanno scritte in modo identico agli M-file, tranne pe
l'intestazione che è del tipo:

```
function [variabili uscita]=nomefunzione(variabili ingresso)

funzione [out1, out2, ...] = nomefunzione (IN1, IN2, ...)
```

N.B.: Le funzioni vanno salvate in un file avente lo stesso nome della funzione.
Tutte le variabili sono locali alla funzione, per cui dopo la sua esecuzione non restano in memoria.

La function viene chiamata da linea di comando con:
`nomefunction(valore_variabile_ingresso)`

Esempio di function

```
% ESEMPIO DI FUNCTION: trasformare la macro di prima in
% una function.
function [Atrasp] = trasposta(A)
Atrasp = A';
```

oppure

```
function trasposta(A)
Atrasp = A';
disp('la trasposta è')
Atrasp
```

```
Atrasp = A';
disp('la trasposta della matrice è')
Atrasp
```

Istruzione `if, else, elseif`

`if` valuta un'espressione logica ed esegue una serie di istruzioni a seconda del valore dell'espressione logica.

```
if espressione logica
   istruzioni
elseif espressione logica
   istruzioni
else
   istruzioni
end
```

Operatori di relazione:

>	maggiore
<	minore
>=	maggiore o uguale
<=	minore o uguale
==	uguale
~=	diverso

Esempio istruzione if (1)

Aprire un file nuovo, salvarlo con nome dispar.m
La function prende in input un numero e controlla se esso è pari o dispari.

```
function dispar(x)
if rem(x,2) == 0
    disp('Il numero è pari')
else
    disp ('Il numero è dispari')
end
```

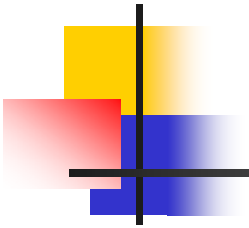
Esempio istruzione if (2)

```
% function per calcolare l'inversa di una matrice 2x2

function [B] = inversa(A)

if (A(1,1)*A(2,2))-(A(1,2)*A(2,1)) == 0
    disp('la matrice non è invertibile')
else
    detA = (A(1,1)*A(2,2))-(A(1,2)*A(2,1));
end

B = 1/detA*[A(2,2), -A(1,2);A(1,1), - A(2,1)];
```



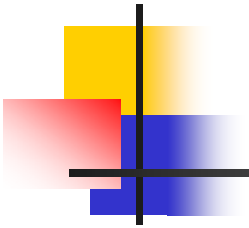
Ciclo `for`

Il ciclo **`for`** esegue un numero di istruzioni per un numero fissato di volte.

```
for indice = inizio:incremento:fine
   istruzioni
end
```

Esempio: somma degli elementi di un vettore

```
somma = 0;
v = [3 6 7 0 3];
for j=1:1:length(v)
     somma = somma + v(j);
end
```



Esempio di ciclo for

```
% Calcolare il valore medio di un vettore.


x = [1 2 3 4 5 6 7];

somma = 0;
for j = 1:1:length(x)
    somma = somma + x(j);
end

media = somma/length(x)
```

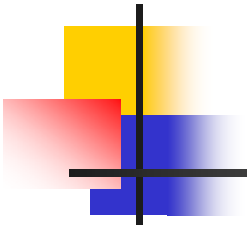
Ciclo while

Il ciclo **while** esegue un numero di istruzioni finché l'espressione di controllo rimane vera.

```
while espressione di controllo
   istruzioni
end
```

Esercizio: dividere un numero per 2 finché il risultato non sia inferiore a 0.005. Contare il numero delle operazioni di divisione effettuate.

```
a = 2390;     % dividendo
b = 2;        % divisore
count = 0;
while (a/b > 0.005)
      c = a/b;
      a = c;
      count = count + 1;
end
count
```



Istruzione switch

switch valuta un'espressione ed esegue un unico caso (**case**) possibile di istruzioni in base al valore di tale espressione.

```
switch espressione_switch
    case espressione_case
      istruzioni,..., istruzioni
    case {espr1, espr2, espr3,...}
      istruzioni,..., istruzioni
...
    otherwise
      istruzioni,..., istruzioni
 end
```

Esempio di istruzione switch

```
% Creare una macro che chiede all'utente di inserire un
% valore di eccentricità e fare visualizzare il tipo
% di conica.
```

Codice per inserire dati da linea di comando:
nomevariabile = input('testo da visualizzare')

```
eccentricita = input('Inserire un valore di eccentricità  ')

switch (eccentricita)
    case eccentricita == 1
        disp('La conica è una parabola!')
    case eccentricita == 0
        disp('La conica è una circonferenza!')
    case eccentricita > 1
        disp('La conica è un iperbole!')
    otherwise
        disp('La conica è un ellisse!')
end
```